REMARKS

This is a full and timely response to the outstanding non-final Office Action mailed November 15, 2007. Claims 1-28 remain pending in the present application. Reconsideration and allowance of the application and pending claims are respectfully requested.

1. Response to Rejections of Claims under 35 U.S.C. § 112

Claim 22 has been rejected under 35 U.S.C. § 112, First Paragraph because the claim consists of a single means and is allegedly subject to an undue breadth rejection. Claim 22 has been amended to address the Examiner's concerns. Therefore, withdrawal of the rejection is respectfully requested.

2. Response to Rejections of Claims under 35 U.S.C. § 101

Claims 27-28 have been rejected under 35 U.S.C. § 101 as allegedly being directed to non-statutory subject matter. In particular, the Office Action states that the claimed means are considered to be software and not associated with any physical structure.

In response, independent claim 27, as presented, recites a computer readable medium that is a physical structure and has a computer program that cause a computer to perform a method.

Applicant notes that it is the view of the Patent Office that:

> [t]he computer readable medium must be physical structure which provides the functional descriptive material in usable form to permit the functionality to be realized with the computer. A program product which does not explicitly include such a medium, a program per se, a signal or other type of transmission media that fails to include the hardware necessary to realize the functionality (e.g., a transmitter or a receiver), and a piece of paper with the functional descriptive material written on it are all examples of media which are not believed to enable the functionality to be realized with the computer.

See http://www.uspto.gov/web/offices/pac/compexam/interim_guide_ subj_ matter_ eligibility.html (Emphasis added). Accordingly, claim 27 as presented complies with applicable patent laws and rules, since it recites a computer readable medium that is a

physical structure which provides functional descriptive material in usable form to permit the functionality to be realized with a computer. Accordingly, withdrawal of the rejection of claims 27-28 is respectfully requested.

3. Response To Rejections of Claims Under 35 U.S.C. § 102

Claims 1-9, 16-17, and 19-28 have been rejected under 35 U.S.C. § 102(e) as allegedly being anticipated by *Apte* (U.S. Patent No. 6,959,307). Applicant respectfully traverses this rejection.

It is axiomatic that "[a]nticipation requires the disclosure in a single prior art reference of each element of the claim under consideration." *W. L. Gore & Associates, Inc. v. Garlock, Inc.*, 721 F.2d 1540, 1554, 220 USPQ 303, 313 (Fed. Cir. 1983). Therefore, every claimed feature of the claimed subject matter must be represented in the applied reference to constitute a proper rejection under 35 U.S.C. § 102(e). In the present case, not every feature of the claimed subject matter is represented in the *Apte* reference. Applicant discusses the *Apte* reference and Applicant's claims in the following.

a. Claim 1

Per claim 1, Applicant claims:

> A remote object invocation method for invoking a method of a remote object of a remote server; the method comprising the steps of:
> ***producing remote object data associated with the remote object to discover an object interface dynamically;***
> ***interpretatively establishing a proxy object of a local client computer using the remote object data at runtime of client software, the proxy object bearing an associated proxy method corresponding to the method of the remote object;***
> invoking, in response to an action of the client software, the proxy object method;
> conveying invocation data associated with the invocation of the proxy method to the remote object;
> ***invoking, in response to the invocation data, the method of the remote object; and***
> ***returning invocation result data to the client software via the proxy object.***

(Emphasis added).

Applicant respectfully submits that independent claim 1 is allowable for at least the reason that *Apte* does not disclose, teach, or suggest at least "producing remote object data associated with the remote object to discover an object interface dynamically," "interpretatively establishing a proxy object of a local client computer using the remote object data at runtime of client software, the proxy object bearing an associated proxy method corresponding to the method of the remote object," "invoking, in response to the invocation data, the method of the remote object," and "returning invocation result data to the client software via the proxy object," as emphasized above.

Rather, *Apte* describes that the "present invention allows a Java client to invoke the business methods of an EJB residing on a CORBA client." Col. 10, lines 40-42. Within this environment, *Apte* describes that an "[o]bject dispatcher 764 passes an object request through data marshalling module 766 to remote call skeleton 768. . . . Remote call skeleton 768 invokes the method in server object 770 and passes the appropriate arguments to server object 770." Col. 11, lines 59-67. "[T]he object reference for the server object is wrapped with an adapter so that the client object's invocation of a method in the server object is transparently handled by an adapter. When the client object attempts to invoke a method of the server object, the method is actually invoked within an adapter object. The adapter class essentially wraps the object reference in a manner such that the object reference is isolated from the client code. The client does not know anything about the object reference. The client only 'talks' to the adapter code whereas the object reference that 'talks' to the skeleton code on the server does not know anything about the client-based adapter." Col. 12, lines 38-49.

In contrast, the claimed subject matter describes "producing remote object data associated with the remote object to discover an object interface dynamically," "interpretatively establishing a proxy object of a local client computer using the remote object data at runtime of client software, the proxy object bearing an associated proxy method corresponding to the method of the remote object," "invoking, in response to the invocation data, the method of the remote object," and "returning invocation result data to the client software via the proxy object." As explained in the present application, "[t]he use of introspection advantageously removes the need to produce manually an

IDL compiled interface description" and "the need to define an IDL and associated skeletons explicitly is eliminated." Page 9, lines 25-26 and page 11, lines 18-20. However, *Apte* discloses the use of skeletons and likewise does not disclose the use of introspection to define an object's interface dynamically. Accordingly, *Apte* fails to teach or suggest at least "producing remote object data associated with the remote object to discover an object interface dynamically," "interpretatively establishing a proxy object of a local client computer using the remote object data at runtime of client software, the proxy object bearing an associated proxy method corresponding to the method of the remote object," "invoking, in response to the invocation data, the method of the remote object," and "returning invocation result data to the client software via the proxy object," as recited in claim 1.

Thus, *Apte* fails to teach or suggest all of the features of claim 1. As a result, claim 1 is not anticipated by *Apte*, and the rejection should be withdrawn.

b. Claims 2-9, 22, and 27-28

Dependent claims 2-9, 22, and 27-28 (which depend from independent claim 1) are allowable as a matter of law for at least the reason that dependent claims 2-9, 22, and 27-28 contain all the features of independent claim 1. For at least this reason, the rejections of claims 2-9, 22, and 27-28 should be withdrawn.

Additionally and notwithstanding the foregoing reasons for allowability of claims 2-9, 22, and 27-28, these claims recite further features and/or combinations of features (as is apparent by examination of the claim itself) that are patentably distinct from the cited art of record. Hence, there are other reasons why these dependent claims are allowable.

c. Claim 16

Per claim 16, Applicant claims:

> A remote object invocation method for invoking a method of a remote object of a remote server; the method comprising the steps of ***introspecting the remote object to produce introspection data associated with the method, the introspection data including an interface description for the remote object; interpretatively***

processing the introspection data to establish a proxy object of a local client computer bearing an associated proxy method for the remote object method; invoking, in response to an action of client software, the proxy object method; conveying invocation data associated with the proxy object method to the remote object; invoking, in response to the invocation data, the method of the remote object; and returning invocation result data to the client software via the object proxy.

(Emphasis added).

Applicant respectfully submits that independent claim 16 is allowable for at least the reason that *Apte* does not disclose, teach, or suggest at least "introspecting the remote object to produce introspection data associated with the method, the introspection data including an interface description for the remote object," "interpretatively processing the introspection data to establish a proxy object of a local client computer bearing an associated proxy method for the remote object method," "invoking, in response to an action of client software, the proxy object method," "conveying invocation data associated with the proxy object method to the remote object," invoking, in response to the invocation data, the method of the remote object," and "returning invocation result data to the client software via the object proxy," as emphasized above.

Rather, *Apte* describes that the "present invention allows a Java client to invoke the business methods of an EJB residing on a CORBA client." Col. 10, lines 40-42. Within this environment, *Apte* describes that an "[o]bject dispatcher 764 passes an object request through data marshalling module 766 to remote call skeleton 768. . . . Remote call skeleton 768 invokes the method in server object 770 and passes the appropriate arguments to server object 770." Col. 11, lines 59-67. "[T]he object reference for the server object is wrapped with an adapter so that the client object's invocation of a method in the server object is transparently handled by an adapter. When the client object attempts to invoke a method of the server object, the method is actually invoked within an adapter object. The adapter class essentially wraps the object reference in a manner such that the object reference is isolated from the client code. The client does not know anything about the object reference. The client only 'talks' to the adapter code whereas the object reference that 'talks' to the skeleton code

on the server does not know anything about the client-based adapter." Col. 12, lines 38-49.

In contrast, the claimed subject matter describes introspecting a remote object to produce introspection data associated with a method, where the introspection data includes an interface description for the remote object. As explained in the present application, "[t]he use of introspection advantageously removes the need to produce manually an IDL compiled interface description" and "the need to define an IDL and associated skeletons explicitly is eliminated." Page 9, lines 25-26 and page 11, lines 18-20. However, *Apte* discloses the use of skeletons and likewise does not disclose the use of introspection to define an object's interface dynamically. Accordingly, *Apte* fails to teach or suggest at least "introspecting the remote object to produce introspection data associated with the method, the introspection data including an interface description for the remote object," "interpretatively processing the introspection data to establish a proxy object of a local client computer bearing an associated proxy method for the remote object method," "invoking, in response to an action of client software, the proxy object method," "conveying invocation data associated with the proxy object method to the remote object," invoking, in response to the invocation data, the method of the remote object," and "returning invocation result data to the client software via the object proxy," as recited in claim 16.

Thus, *Apte* fails to teach or suggest all of the features of claim 16. As a result, claim 16 is not anticipated by *Apte*, and the rejection should be withdrawn.

d. Claim 17

Per claim 17, Applicant claims:

> A method of invoking, from a first computer, a remote object located on a second computer, the method comprising the steps of ***introspecting, at the second computer, the remote object to identify at least one of a method, property and event thereof to produce introspection data that describes any such identified methods, properties and events; transmitting the introspection data, via a first transport mechanism, to the first computer; creating, at the first computer, a proxy object from the introspection data; invoking, at the first computer, a method of the proxy object; transmitting, from the first computer to the second***

> ***computer, remote method invocation data, via a second transport mechanism, the remote method invocation data comprising at least an indication of the remote object and the method, property and event thereof to be invoked;*** receiving, at the second computer, via the second transport mechanism, the remote method invocation data, extracting, at the second computer, the remote method invocation data from the second transport data structure; invoking, at the second computer, the method, property or event of the remote object identified by the remote object invocation data; transmitting, from the second computer to the first computer, via a third transport mechanism, a return object or data representing the results of the invocation of the method, property or event of the remote object; and extracting, at the first computer, the return object or data from the third transport mechanism; wherein the step of creating, at the first computer, the proxy object from the introspection data comprises the step of interpretatively parsing the introspection data and interpretatively constructing the proxy object.

(Emphasis added).

Applicant respectfully submits that independent claim 17 is allowable for at least the reason that *Apte* does not disclose, teach, or suggest at least "introspecting, at the second computer, the remote object to identify at least one of a method, property and event thereof to produce introspection data that describes any such identified methods, properties and events; transmitting the introspection data, via a first transport mechanism, to the first computer; creating, at the first computer, a proxy object from the introspection data; invoking, at the first computer, a method of the proxy object; [and] transmitting, from the first computer to the second computer, remote method invocation data, via a second transport mechanism, the remote method invocation data comprising at least an indication of the remote object and the method, property and event thereof to be invoked," as emphasized above.

Rather, *Apte* describes that the "present invention allows a Java client to invoke the business methods of an EJB residing on a CORBA client." Col. 10, lines 40-42. Within this environment, *Apte* describes that an "[o]bject dispatcher 764 passes an object request through data marshalling module 766 to remote call skeleton 768. . . . Remote call skeleton 768 invokes the method in server object 770 and passes the appropriate arguments to server object 770." Col. 11, lines 59-67. "[T]he object reference for the server object is wrapped with an adapter so that the client object's invocation of a method in the server object is transparently handled by an adapter.

When the client object attempts to invoke a method of the server object, the method is actually invoked within an adapter object. The adapter class essentially wraps the object reference in a manner such that the object reference is isolated from the client code. The client does not know anything about the object reference. The client only 'talks' to the adapter code whereas the object reference that 'talks' to the skeleton code on the server does not know anything about the client-based adapter." Col. 12, lines 38-49.

In contrast, the claimed subject matter describes introspecting, at a second computer, a remote object to identify at least one of a method, property and event thereof to produce introspection data that describes any such identified methods, properties and events. As explained in the present application, "[t]he use of introspection advantageously removes the need to produce manually an IDL compiled interface description" and "the need to define an IDL and associated skeletons explicitly is eliminated." Page 9, lines 25-26 and page 11, lines 18-20. However, *Apte* discloses the use of skeletons and likewise does not disclose the use of introspection to define an object's interface dynamically. Accordingly, *Apte* fails to teach or suggest at least "introspecting, at the second computer, the remote object to identify at least one of a method, property and event thereof to produce introspection data that describes any such identified methods, properties and events; transmitting the introspection data, via a first transport mechanism, to the first computer; creating, at the first computer, a proxy object from the introspection data; invoking, at the first computer, a method of the proxy object; [and] transmitting, from the first computer to the second computer, remote method invocation data, via a second transport mechanism, the remote method invocation data comprising at least an indication of the remote object and the method, property and event thereof to be invoked," as recited in claim 17.

Thus, *Apte* fails to teach or suggest all of the features of claim 17. As a result, claim 17 is not anticipated by *Apte*, and the rejection should be withdrawn.

e. **Claim 19**

Per claim 19, Applicant claims:

> A data processing method to invoke a method of a remote object of a remote server; the method comprising the steps of ***producing remote object data associated with the remote object to discover an object interface dynamically; and interpretatively establishing a proxy object at a local client computer using the remote object data at runtime of client software invoking an associated proxy object method, the proxy object bearing the associated proxy method corresponding to the method of the remote object.***

(Emphasis added).

Applicant respectfully submits that independent claim 19 is allowable for at least the reason that *Apte* does not disclose, teach, or suggest at least "producing remote object data associated with the remote object to discover an object interface dynamically; and interpretatively establishing a proxy object at a local client computer using the remote object data at runtime of client software invoking an associated proxy object method, the proxy object bearing the associated proxy method corresponding to the method of the remote object," as emphasized above.

Rather, *Apte* describes that the "present invention allows a Java client to invoke the business methods of an EJB residing on a CORBA client." Col. 10, lines 40-42. Within this environment, *Apte* describes that an "[o]bject dispatcher 764 passes an object request through data marshalling module 766 to remote call skeleton 768. . . . Remote call skeleton 768 invokes the method in server object 770 and passes the appropriate arguments to server object 770." Col. 11, lines 59-67. "[T]he object reference for the server object is wrapped with an adapter so that the client object's invocation of a method in the server object is transparently handled by an adapter. When the client object attempts to invoke a method of the server object, the method is actually invoked within an adapter object. The adapter class essentially wraps the object reference in a manner such that the object reference is isolated from the client code. The client does not know anything about the object reference. The client only 'talks' to the adapter code whereas the object reference that 'talks' to the skeleton code on the server does not know anything about the client-based adapter." Col. 12, lines 38-49.

In contrast, the claimed subject matter describes producing remote object data associated with the remote object to discover an object interface dynamically and interpretatively establishing a proxy object at a local client computer using the remote object data at runtime of client software invoking an associated proxy object method. As explained in the present application, "[t]he use of introspection advantageously removes the need to produce manually an IDL compiled interface description" and "the need to define an IDL and associated skeletons explicitly is eliminated." Page 9, lines 25-26 and page 11, lines 18-20. However, *Apte* discloses the use of skeletons and likewise does not disclose the use of introspection to define an object's interface dynamically. Accordingly, *Apte* fails to teach or suggest at least "producing remote object data associated with the remote object to discover an object interface dynamically; and interpretatively establishing a proxy object at a local client computer using the remote object data at runtime of client software invoking an associated proxy object method, the proxy object bearing the associated proxy method corresponding to the method of the remote object," as recited in claim 19.

Thus, *Apte* fails to teach or suggest all of the features of claim 19. As a result, claim 19 is not anticipated by *Apte*, and the rejection should be withdrawn.

f. Claims 20-21

Dependent claims 20-21 (which depend from independent claim 19) are allowable as a matter of law for at least the reason that dependent claims 20-21 contain all the features of allowable independent claim 19. For at least this reason, the rejections of claims 20-21 should be withdrawn.

Additionally and notwithstanding the foregoing reasons for allowability of claims 20-21, these claims recite further features and/or combinations of features (as is apparent by examination of the claim itself) that are patentably distinct from the cited art of record. Hence, there are other reasons why these dependent claims are allowable.

g. Claim 23

Per claim 23, Applicant claims:

> A remote object invocation system to invoke a method of a remote object of a remote server; the system comprising a local computer, ***wherein the local computer comprises an inspector to produce remote object data associated with the remote object, the remote object data including an interface description for the remote object; an interpreter to interpretatively establishing a proxy object using the remote object data at runtime of client software of the local computer; the proxy object bearing an associated proxy method corresponding to the method of remote object; a local invocation means to invoke, in response to an action of the client software, the proxy object method; and a carrier to convey invocation data associated with the invocation of the proxy method to the remote object***.

(Emphasis added).

Applicant respectfully submits that independent claim 23 is allowable for at least the reason that *Apte* does not disclose, teach, or suggest at least "wherein the local computer comprises an inspector to produce remote object data associated with the remote object, the remote object data including an interface description for the remote object; an interpreter to interpretatively establishing a proxy object using the remote object data at runtime of client software of the local computer; the proxy object bearing an associated proxy method corresponding to the method of remote object; a local invocation means to invoke, in response to an action of the client software, the proxy object method; and a carrier to convey invocation data associated with the invocation of the proxy method to the remote object," as recited and emphasized above in claim 23.

Rather, *Apte* describes that the "present invention allows a Java client to invoke the business methods of an EJB residing on a CORBA client." Col. 10, lines 40-42. Within this environment, *Apte* describes that an "[o]bject dispatcher 764 passes an object request through data marshalling module 766 to remote call skeleton 768. . . . Remote call skeleton 768 invokes the method in server object 770 and passes the appropriate arguments to server object 770." Col. 11, lines 59-67. "[T]he object reference for the server object is wrapped with an adapter so that the client object's invocation of a method in the server object is transparently handled by an adapter.

When the client object attempts to invoke a method of the server object, the method is actually invoked within an adapter object. The adapter class essentially wraps the object reference in a manner such that the object reference is isolated from the client code. The client does not know anything about the object reference. The client only 'talks' to the adapter code whereas the object reference that 'talks' to the skeleton code on the server does not know anything about the client-based adapter." Col. 12, lines 38-49.

In contrast, the claimed subject matter describes a local computer producing remote object data associated with a remote object, where the remote object data includes an interface description for the remote object and an interpreter to interpretatively establishing a proxy object using the remote object data at runtime of client software of the local computer. As explained in the present application, "[t]he use of introspection advantageously removes the need to produce manually an IDL compiled interface description" and "the need to define an IDL and associated skeletons explicitly is eliminated." Page 9, lines 25-26 and page 11, lines 18-20. However, *Apte* discloses the use of skeletons and likewise does not disclose the use of introspection to define an object's interface dynamically. Accordingly, *Apte* fails to teach or suggest at least "wherein the local computer comprises an inspector to produce remote object data associated with the remote object, the remote object data including an interface description for the remote object; an interpreter to interpretatively establishing a proxy object using the remote object data at runtime of client software of the local computer; the proxy object bearing an associated proxy method corresponding to the method of remote object; a local invocation means to invoke, in response to an action of the client software, the proxy object method; and a carrier to convey invocation data associated with the invocation of the proxy method to the remote object," as recited in claim 23.

Thus, *Apte* fails to teach or suggest all of the features of claim 23. As a result, claim 23 is not anticipated by *Apte*, and the rejection should be withdrawn.

h. Claim 24

Dependent claim 24 (which depends from independent claim 23) is allowable as a matter of law for at least the reason that dependent claim 24 contains all the features of allowable independent claim 23. For at least this reason, the rejection of claim 24 should be withdrawn.

i. Claim 25

Per claim 25, Applicant claims:

> A ***remote object server*** hosting a remote object; the server comprising ***means to generate introspection data associated with the remote object, the introspection data including an interface description for the remote object and means to output the introspection data for use by a client in interpretatively creating a proxy object using the introspection data***.

(Emphasis added).

Applicant respectfully submits that independent claim 25 is allowable for at least the reason that *Apte* does not disclose, teach, or suggest at least "means to generate introspection data associated with the remote object, the introspection data including an interface description for the remote object and means to output the introspection data for use by a client in interpretatively creating a proxy object using the introspection data," as emphasized above.

Rather, *Apte* describes that the "present invention allows a Java client to invoke the business methods of an EJB residing on a CORBA client." Col. 10, lines 40-42. Within this environment, *Apte* describes that an "[o]bject dispatcher 764 passes an object request through data marshalling module 766 to remote call skeleton 768. . . . Remote call skeleton 768 invokes the method in server object 770 and passes the appropriate arguments to server object 770." Col. 11, lines 59-67. "[T]he object reference for the server object is wrapped with an adapter so that the client object's invocation of a method in the server object is transparently handled by an adapter. When the client object attempts to invoke a method of the server object, the method is actually invoked within an adapter object. The adapter class essentially wraps the object reference in a manner such that the object reference is isolated from the client

code. The client does not know anything about the object reference. The client only 'talks' to the adapter code whereas the object reference that 'talks' to the skeleton code on the server does not know anything about the client-based adapter." Col. 12, lines 38-49.

In contrast, the claimed subject matter describes generating introspection data associated with the remote object, where the introspection data includes an interface description for the remote object. As explained in the present application, "[t]he use of introspection advantageously removes the need to produce manually an IDL compiled interface description" and "the need to define an IDL and associated skeletons explicitly is eliminated." Page 9, lines 25-26 and page 11, lines 18-20. However, *Apte* discloses the use of skeletons and likewise does not disclose the use of introspection to define an object's interface dynamically. Accordingly, *Apte* fails to teach or suggest at least "means to generate introspection data associated with the remote object, the introspection data including an interface description for the remote object and means to output the introspection data for use by a client in interpretatively creating a proxy object using the introspection data," as recited in claim 25.

Thus, *Apte* fails to teach or suggest all of the features of claim 25. As a result, claim 25 is not anticipated by *Apte*, and the rejection should be withdrawn.

j. Claim 26

Per claim 26, Applicant claims:

> A ***client computer*** hosting a receiver for ***receiving introspection data associated with a remote object of a remote server, the introspection data including an interface description for the remote object, and an interpreter for interpretatively creating a proxy object, using the introspection data, for invocation by an application executable at the client computer***.

(Emphasis added).

Applicant respectfully submits that independent claim 26 is allowable for at least the reason that *Apte* does not disclose, teach, or suggest at least "receiving introspection data associated with a remote object of a remote server, the introspection data including an interface description for the remote object, and an interpreter for

interpretatively creating a proxy object, using the introspection data, for invocation by an application executable at the client computer," as emphasized above.

Rather, *Apte* describes that the "present invention allows a Java client to invoke the business methods of an EJB residing on a CORBA client." Col. 10, lines 40-42. Within this environment, *Apte* describes that an "[o]bject dispatcher 764 passes an object request through data marshalling module 766 to remote call skeleton 768. . . . Remote call skeleton 768 invokes the method in server object 770 and passes the appropriate arguments to server object 770." Col. 11, lines 59-67. "[T]he object reference for the server object is wrapped with an adapter so that the client object's invocation of a method in the server object is transparently handled by an adapter. When the client object attempts to invoke a method of the server object, the method is actually invoked within an adapter object. The adapter class essentially wraps the object reference in a manner such that the object reference is isolated from the client code. The client does not know anything about the object reference. The client only 'talks' to the adapter code whereas the object reference that 'talks' to the skeleton code on the server does not know anything about the client-based adapter." Col. 12, lines 38-49.

In contrast, the claimed subject matter describes receiving introspection data associated with a remote object of a remote server, where the introspection data includes an interface description for the remote object and an interpreter for interpretatively creating a proxy object, using the introspection data, for invocation by an application executable at the client computer, as described in claim 26. As explained in the present application, "[t]he use of introspection advantageously removes the need to produce manually an IDL compiled interface description" and "the need to define an IDL and associated skeletons explicitly is eliminated." Page 9, lines 25-26 and page 11, lines 18-20. However, *Apte* discloses the use of skeletons and likewise does not disclose the use of introspection to define an object's interface dynamically. Accordingly, *Apte* fails to teach or suggest at least "receiving introspection data associated with a remote object of a remote server, the introspection data including an interface description for the remote object, and an interpreter for interpretatively

creating a proxy object, using the introspection data, for invocation by an application executable at the client computer," as recited in claim 26.

Thus, *Apte* fails to teach or suggest all of the features of claim 26. As a result, claim 26 is not anticipated by *Apte*, and the rejection should be withdrawn.

4. Response to Rejections of Claims under 35 U.S.C. § 103

In the Office Action, claims 10-15 and 18 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over *Apte* in view of *Tewksbary* (U.S. Patent Application Publication No. 2004/0210585).

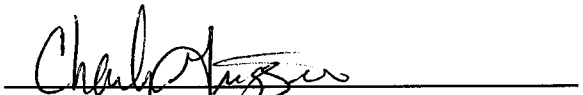
Dependent claims 10-15 (which depend from independent claim 1) are allowable as a matter of law for at least the reason that dependent claims 10-15 contain all the features of independent claim 1 and *Tewksbary* fails to cure the deficiencies of the *Apte* reference.

Likewise, because independent claim 17 is allowable over the cited art of record, dependent claim 18 (which depends from independent claim 17) is allowable as a matter of law for at least the reason that dependent claim 18 contains all the features of independent claim 17 and *Tewksbary* fails to cure the deficiencies of the *Apte* reference.

CONCLUSION

For at least the reasons set forth above, Applicant respectfully submits that all objections and/or rejections have been traversed, rendered moot, and/or accommodated, and that the pending claims are in condition for allowance. Favorable reconsideration and allowance of the present application and all pending claims are hereby courteously requested. If, in the opinion of the Examiner, a telephonic conference would expedite the examination of this matter, the Examiner is invited to call the undersigned agent at (770) 933-9500.

Respectfully submitted,



Charles W. Griggers, Reg. No. 47,283